Development of the Raw Material Analysis Tool

By

Victor Wai Han Chan

A thesis submitted to the Johns Hopkins University in conformity with the requirements for Master of Science in Engineering

Baltimore, Maryland

May, 2019

Abstract

A Computational Project to develop a tool for the analysis of raw material data was proposed and completed at GlaxoSmithKline. Using Statistica Visual Basic (SVB), a script exceeding 75,000 lines of code and spanning two files, was developed. The code was designed for the purpose aggregating raw material and process data available from previous GlaxoSmithKline projects (or campaigns), as well as generating new predictions based on existing material lots available and the possible ways in which these raw materials can be deployed. The script is designed to be user-friendly, and able to function with minimal user input by automating every step of the analysis and process. Development of the Raw Material Analysis Tool (Analysis Tool) generated new insights into how new lots of raw materials can be used for the current project to maximize process efficiency.

Primary Reader: Marc Donohue

Secondary Reader: Joelle Frechette

Acknowledgements

I would like to dedicate my thesis to the following people:

Brian Gray, my supervisor, for his support and guidance throughout the project.

Arthur Edge, who provided useful feedback throughout the course of the project.

Professor Marc Donohue, for keeping my project on track with the academic requirements of the Department.

Luke Thorstenson and Camille Mathis, for their guidance with the INBT Co-Op.

Table of Contents

List of Figures

1. About GlaxoSmithKline

GlaxoSmithKline (GSK) is a British Pharmaceutical Company headquartered in London. GSK is one of the largest pharmaceutical companies in the world, and is also ranked in the Fortune Global 500, which ranks the largest companies in the world by revenue. Top brands from GSK include Sensodyne Toothpaste.

GSK has offices and facilities in over a hundred countries, and employs almost a hundred thousand people. Four GSK scientists have received Nobel Prizes for their contributions to medical research. Additionally, GSK has also led the Global Access to Medicines Index, which evaluates pharmaceutical companies on their efforts to improve access to medicines.

2. Introduction

Statistics and Regression Models play an important role in Pharmaceutical research, and have been used in a wide range of studies from evaluating formulations (Amraei and Niazi, 2018) to analyzing product sales (Rentala and Anand, 2014). Statistical analyses reduce the need for raw materials and equipment usage (Alin et al., 2019), and enable predictions to be made by identifying correlations in a set of experimental data.

Process optimization in pharmaceutical manufacturing is rarely uncomplicated. A large number of variables, from cell count to mass, could potentially be an important factor. Due to the large numbers of variables involved, which could potentially be of different types (for example, categorical and continuous variables), elementary linear regression is mostly insufficient to capture the full complexity of the data provided. Hence, Multivariate Statistical methods, such as Principal

Component Analysis (PCA), are often an important part of such analyses. Multivariate Statistics can be used to determine how a diverse set of variables could correlate with each other (Lewis et al., 2018). Additionally, Multivariate statistical methods can be used to condense a large set of diverse variables into one or a smaller number of variables (Wang et al., 2018) for dimensional reduction. Many previous studies have indicated the effectiveness of Multivariate models in studying drug formulations (Amraei and Niazi, 2018) and increasing process efficiency (Wang et al., 2018).

GlaxoSmithKline's current project involves the optimization of process variables by determining possible correlations with the composition of raw materials used, and making adjustments accordingly. Previous projects have provided a set of experimental data indicating the performance of different process variables under varying feed and media combinations. The currently available experimental data forms the foundation of the project. Due to the complexity of the composition of the raw materials, including over a hundred different elements, amino acids and vitamins, Multivariate statistical models will be used to capture the complexity of the data and determine possible correlations with different process variables. The objective of the project would be to develop a Multivariate Statistical Model, using one of a selection of Programming/Statistical computer applications available to GlaxoSmithKline, to determine the ideal feed and media quantities to optimize the chosen process variables.

3. GSK Raw Material Analysis Tool

3.1 Overview

The Raw Material Analysis Tool (Analysis Tool) was developed in Statistica Visual Basic (SVB) to provide a rapid and automated solution to analyzing large quantities of raw material data, as well as providing information for the current project. Data from previous projects are analyzed, and the Analysis Tool provides process predictions for multiple selections of raw materials available for the current project.

Operation of the Analysis Tool is intended to be as convenient and easy as possible. Through scripts to load spreadsheets, the tool prompts users to select an excel file for aggregation and analysis. Once loaded, the Tool extracts relevant data from the spreadsheets and provides a list of options for the user. These options are covered in subsequent sections.

For the purpose of company confidentiality, information may be censored in code examples.

3.2 Software Selection

The project requires the development of a statistical model, provided with an executable script or executable capable of performing all tasks without any complex user inputs. While most calculations could technically be performed by hand in conventional data management applications, such as Microsoft Excel, the model needs to be able to automate all these steps. This reduces the manpower cost of each analysis, and also eliminates human error. Additionally, the model should have some basic analytical capabilities of its own, in order to filter unimportant variables and identify useful trends.

The model should ideally be easy to use and available to GlaxoSmithKline staff. As most data is provided through spreadsheets, the accompanying script or executable needs to be capable of breaking down the provided data without complex user inputs.

Statistica, along with Statistica Visual Basic (not to be confused with Visual Basic by Microsoft), was chosen for the project after evaluating all requirements. Statistica is available to GlaxoSmithKline staff, and comes with spreadsheets and a large selection of statistical tools, including multivariate analysis. Statistica Visual Basic provides a simple and efficient programming language which can be used to develop an accompanying script for the statistical model, hence allowing the full process to be automated with a single push of the "run" button.

3.3 Data Aggregation

Experimental data (from previous projects) have been provided in five spreadsheets. They are as follows. For non-disclosure purposes, the exact names of the spreadsheets are not included, and descriptions are generalized to protect company confidentiality.

1) Composition Data – the composition of raw materials by batch and type, based on constituent amino acids, vitamins and elements
2) Quantity Data – the quantity of raw materials used in each run
3) Process Data I – Process variables by each run
4) Process Data II – Process variables by each run
5) Process Data III – Process variables by each run

During the previous project, experiments were conducted in runs, and assigned a unique code. Due to the nature of existing data collection systems, data is provided in the disparate format of

separate spreadsheets. Hence, there is a need to aggregate data in the system. The Analysis Tool starts off with a simple data aggregation system, which sorts and organizes data into a single spreadsheet, while keeping important data available internally in the program for further analysis and regression.

The initial step is identifying unique codes and comparing them across all spreadsheets. Using the first Process Data File, the script identifies all codes (while removing and warning the user of any repeat entries). Subsequently, each code is individually compared against all spreadsheets (except Composition Data) to match all data sets together. Material lot numbers are provided in the Composition Data Spreadsheet, which matched to lot numbers in the Quantity Data Spreadsheet, and subsequently linked to the Process Data Spreadsheets. The script discards any codes which fail to yield connecting data to all uploaded spreadsheets, and also warns the user. The script must find at least one set of usable codes to proceed with data aggregation, although it should be noted that subsequent statistical analyses will require a large number of useable codes for greater accuracy, as each code represents a data point which will be used subsequently.

Aggregating the data in all five spreadsheets is rarely a linear process. Each code (based on each unique run) could be attached to one or more material batches in variable quantities. Single values for elemental, amino acid and vitamin compositions are required for each code, by calculating proportional contributions from each batch.

Data aggregation has largely been accomplished through the use of arrays and loops. Arrays are a data structure consisting of a set of elements, which could be integers, text and so on. Unless otherwise mentioned, all arrays used are one-dimensional. For each code, the script runs through all matching batch numbers and stores them in an array. This array is reused for each code, allowing to minimize the generation of arrays. This allows for efficient management of data. A

similar method is also applied to calculating feed and media compositions by code. For each code, arrays store information (composition and mass) for each related material batch, and consolidates the information in a single scripted calculation.

```
l = 0
m = 0
While l < CasNumMEs
    l = l + 1
    If StrComp(CompressCodeSet(k,1),FinalBatchME(l,1)) =0 And RawMaterialDescriptionME(l,1) Like ""[illegible]"" Then
        If HierarchyME(l,1) Like ""[illegible]"" Or HierarchyME(l,1) Like ""[illegible]"" Then
            If m < 1 Then
                m = m + 1
                StringArrayHolder(m,1) = RawMaterialBatchNoME(l,1)
            Else
                n = 0
                EStop = 0
                While n < m
                    n = n + 1
                    If StrComp(StringArrayHolder(n,1), RawMaterialBatchNoME(l,1)) = 0 Then
                        EStop = 11
                    End If
                Wend
                If EStop < 10 Then
                    m = m + 1
                    StringArrayHolder(m,1) = RawMaterialBatchNoME(l,1)
                End If
            End If
        End If
    End If
Wend
n = 1
RMBN[illegible] = StringArrayHolder(n,1)
While n < m
    n = n + 1
    RMBN[illegible] = RMBN[illegible] + SLASH + StringArrayHolder(n,1)
Wend
Final.SetData(i,5,RMBN[illegible])
[illegible]SortSheet.SetData(k,5,RMBN[illegible])
```

Figure 1: SVB code to identify and list relevant batch values without repetition.

An example is provided in Figure 1. The script first filters irrelevant spreadsheet entries by searching for data which corresponds to a certain raw material type (RawMaterialDescriptionME) and process unit number (HierarchyME). Subsequently, the script adds the corresponding material batch number to a storage array (StringArrayHolder). If the array is empty, the batch number is added automatically. Otherwise, the script runs through all batch numbers already in the array and only adds the batch number if it has yet to do so.

```
l = 0
While l < CasNumMEs
    l = l + 1
    If StrComp(CompressCodeSet(k,1).FinalBatchME(l,1)) =0 And RawMaterialDescriptionME(l,1) Like ""[illegible]"" Then
        Vlocate[illegible](l,1) = 100
        Vvalue[illegible](l,1) = ValueME(l,1)
        Vlot[illegible](l,1) = VendorBatchNoME(l,1)
        Vlot[illegible](l,2) = RawMaterialBatchNoME(l,1)
    End If
Wend
V[illegible] = 0
l = 0
While l < CasNumMEs
    l = l + 1
    If Vlocate[illegible](l,1) > 10 Then
        V[illegible] = V[illegible] + Vvalue[illegible](l,1)
    End If
Wend
Final.SetData(i,7,V[illegible])
[illegible]SortSheet.SetData(k,7,V[illegible])
```

Figure 2: SVB code to mark relevant raw material locations in a spreadsheet and extract identification data.

Figure 2 identifies areas in a spreadsheet where codes align, as well as filtering for data which corresponds to a certain raw material type. When all the conditions are met, a binary array is marked with a value of 100 to indicate that the relative positioning in the associated spreadsheet contains relevant information. Mass values are obtained and stored in another array, while a two-dimensional array stores all important batch numbers, all in an array position corresponding to the original spreadsheet. Following that, a second loop calculates the mass value contribution of the raw material and code, and loads it into the final spreadsheet.

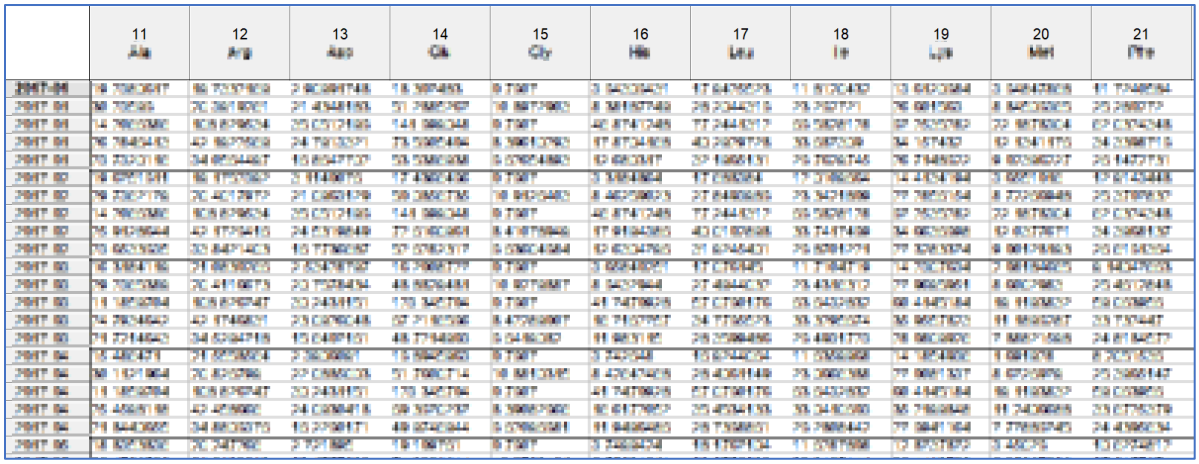
```
e = 0
While e < [illegible]
    e = e + 1
    ex = e + 10
    l = 0
    Ele[illegible] = 0
    While l < CasNumMEs
        l = l + 1
        If Vlocate[illegible](l,1) > 10 Then
            n = 0
            m = 0
            While m < CasNumEDAs
                m = m + 1
                If StrComp(VendorLotNoEDA(m,1),Vlo[illegible](l,1))=0 And StrComp(GSKLotNoEDA(m,1),Vlot[illegible](l,2))=0 Then
                    If n > 0 Then
                        If e = 1 Then
                            [illegible]
                            ErrorActive = 100
                            ErrorString = ErrorString + "Warning: Repeat GSK/Vendor Lot Number in [illegible] for " + GSKLotNoEDA(m,1) + S
                        End If
                    Else
                        Ele[illegible] = Ele[illegible] + (Elemental[illegible](m,e))*(ValueME(l,1))/(V[illegible])
                        n = n + 1
                    End If
                End If
            Wend
            If n = 0 And e = 1 Then
                ErrorActive = 100
                ErrorString = ErrorString + "Warning: No corresponding entry found in [illegible] for [illegible] Number for
            End If
        End If
    Wend
```

Figure 3: SVB code to calculate averaged element, amino acid and vitamin compositions based on proportional contributions from varying batches.

Figure 3 calculates final compositions of each code (from each run) based on proportional contributions from each raw material batch number. Using data from the previous code snippet, relevant data sets are obtained by comparing batch numbers to the previously created array. For each relevant set, a proportional contribution is calculated by taking the sum of all relevant elemental contributions, multiplied by the fractional mass value of each relevant batch number over the total mass used. Data is populated in the spreadsheets, while error checkers warn the user of any gaps or other errors in the data.

Figure 4: Output Spreadsheet of Aggregated Data.

Data is organized into a single spreadsheet (Figure 4) which provides a clean overview of the data with proportional contributions by each batch calculated.



	11 Ala	12 Arg	13 Asn	14 Cit	15 Cly	16 His	17 Leu	18 Ile	19 Lys	20 Met	21 Phe
2007_00											
2007_01											
2007_01											
2007_01											
2007_01											
2007_02											
2007_02											
2007_02											
2007_02											
2007_02											
2007_03											
2007_03											
2007_03											
2007_03											
2007_03											
2007_04											
2007_04											
2007_04											
2007_04											
2007_04											
2007_05											

Figure 5: Output spreadsheet of averaged compositions of feed and media used in process runs.

Compositions by code (in Elements, Amino Acids and Vitamins) are also organized into the spreadsheet, as shown in Figure 5.

3.4 PCA Analysis

Principal Component Analysis (PCA) plays an important role in multivariate analyses, and can provide useful insights in a study with a large number of observed variables. Broadly described, PCA is a dimensional-reduction technique (Ruvalcaba-López et al., 2019), reducing a complex dataset into a simpler one. In the case of this study, PCA condenses a large number of variables and summarizes the data in the form of a smaller number of components, where each component is a linear combination of variables, which, when all components are taken together, can account for the variation of the data as a whole (Felipe-Sotelo et al., 2008).

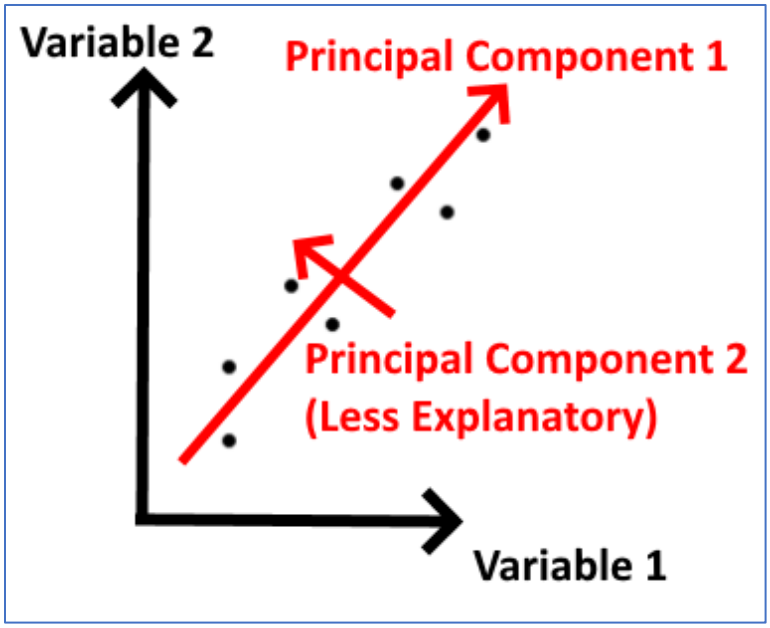


Figure 6: Illustration of PCA Analysis.

Statistica provides an in-built tool to perform Multivariate Analyses, including PCA. The analysis can be performed either through the interface, or using a script, although the tool will make use of the latter.

```
Dim newanalysisPCA As Analysis
Set newanalysisPCA = Analysis (scMSPC, SortSheet)
Dim oStaDocsPCA As StaDocuments

' Multivariate Statistical Process Control: in Final Workbook
Dim oAD1PCA As STAMSPC.MSPCStartup
Set oAD1PCA = newanalysisPCA.Dialog
oAD1PCA.TypeOfAnalysis = scMspcPrincipalComponentAnalysis

newanalysisPCA.Run

' PCA: in Final Workbook
Dim oAD2PCA As STAMSPC.PCAStartup
Set oAD2PCA = newanalysisPCA.Dialog
oAD2PCA.Variables = "
oAD2PCA.MaxIterations = 50
oAD2PCA.EpsilonOfConvergence = 0.0001
oAD2PCA.AutoFitByCrossValidation = True
oAD2PCA.VFoldCrossValidation = True
oAD2PCA.NumberOfCrossValidationFolds = 7
oAD2PCA.RandomSeedOfCrossValidation = 1554921
oAD2PCA.MinPercentageOfValidCases = 80
oAD2PCA.MinPercentageOfValidVars = 80
oAD2PCA.AutoRemoveVariables = True
oAD2PCA.AutoRemoveCases = True
oAD2PCA.ApplyMemoryLimit = False
oAD2PCA.DoNotScaleVariables = False
oAD2PCA.UnitStandardDeviations = True
oAD2PCA.MaxNumberOfVariableBlocks = 10
oAD2PCA.UseAnalysisAndTestSample = False
```
Figure 7: SVB Code to perform a PCA Analysis in Statistica.

```
newanalysisPCA[illegible].UseGlobalOutputSettings = False
With newanalysisPCA[illegible].OutputOption
    .Placement = scAnalysisWorkbook
    .AutoPlaceResultInWorkbook = True
    .PlaceNewResultAtTop = True
    .ReportPlacement = scNoReport
    .ReportFont = "Courier New"
    .ReportFontSize = 9
    .SupplementaryInfoLevel = scInfoLevelNone
    .MSWordPlacement = scNoMSWord
End With

newanalysisPCA[illegible].Run

' PCA Results: [illegible] in Final Workbook
Dim oAD3PCA[illegible] As STAMSPC.PCAResults
Set oAD3PCA[illegible] = newanalysisPCA[illegible].Dialog
oAD3PCA[illegible].SortVariablesByImportance = True
oAD3PCA[illegible].TSquareControlLimit = 99
oAD3PCA[illegible].TSquareWarning = False
oAD3PCA[illegible].TSquareWarningLimit = 95
'oAD3PCA[illegible].SelectedFirstListOfComponents = "1"
'oAD3PCA[illegible].SelectedSecondListOfComponents = "2"
oAD3PCA[illegible].LabelWithNames = True
oAD3PCA[illegible].ComputeScoreLimitFromStdDev = True
oAD3PCA[illegible].ScoreControlLimit = 3
oAD3PCA[illegible].ScoreWarning = False
oAD3PCA[illegible].ScoreWarningLimit = 2
oAD3PCA[illegible].TypeOfControlChart = scMspcShewhartIndividual
oAD3PCA[illegible].UsingUserDefinedTarget = False
oAD3PCA[illegible].UsingUserDefinedStdDev = False
```

Figure 8: SVB Code to establish further conditions in performing a PCA Analysis in Statistica.

The script used to generate a PCA analysis is shown in Figure 7, while setting adjustments to the PCA analysis are shown in Figure 8. Parameters such as variable locations are specified in the script. Other settings, such as font sizes and variable sorting are also accounted for in the script. The original codes can be generated using the "Record Macro" function in Statistica Visual Basic, which generates codes for analyses performed using the software tools.

The analysis tool performs a PCA analysis on a selected set of predictive variables, namely some combination of the elements, amino acids and vitamins which form the composition of each run (organized by code). Information on each PCA analysis is provided in graphs for each data set.

```
Set oStaDocsPCA = oAD3PCA.SummaryView
Set AnalysisOutputPCA = newanalysisPCA.RouteOutput(oStaDocsPCA)
If (AnalysisOutputPCA.HasWorkbook=True) Then
    Set w=AnalysisOutputPCA.Workbook
    Set wi=w.Root.Child
    While (wi.Type<>scWorkbookItemTypeGraph)
        Set wi=wi.Child
    Wend
    Set PCASummary =wi.Extract(scWorkbookExtractCopy)
End If

Set oStaDocsPCA = oAD3PCA.GraphOfImportanceOfPredictors
Set AnalysisOutputPCA = newanalysisPCA.RouteOutput(oStaDocsPCA)
If (AnalysisOutputPCA.HasWorkbook=True) Then
    Set w=AnalysisOutputPCA.Workbook
    Set wi=w.Root.Child
    While (wi.Type<>scWorkbookItemTypeGraph)
        Set wi=wi.Child
    Wend
    Set PCAVarImpt =wi.Extract(scWorkbookExtractCopy)
End If

Set oStaDocsPCA = oAD3PCA.TSquareChart
Set AnalysisOutputPCA = newanalysisPCA.RouteOutput(oStaDocsPCA)
If (AnalysisOutputPCA.HasWorkbook=True) Then
    Set w=AnalysisOutputPCA.Workbook
    Set wi=w.Root.Child
    While (wi.Type<>scWorkbookItemTypeGraph)
        Set wi=wi.Child
    Wend
    Set PCAHotellingT2Control =wi.Extract(scWorkbookExtractCopy)
End If
```

Figure 9: SVB Code to extract graphical information from a PCA Analysis.

Scripts used to call and generate different graphs from the analysis are shown in Figure 9. These are extracted from a temporary workbook created by the analysis and assigned a variable name.

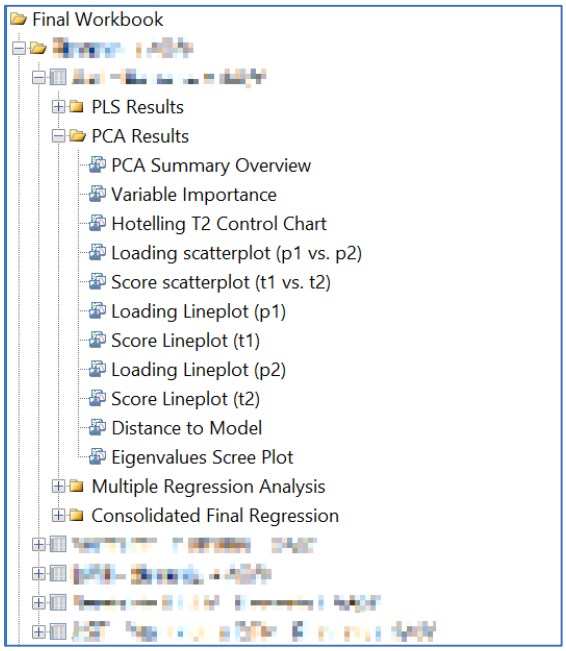


Figure 10: PCA Output graphs from Analysis Tool.

A subset of all available PCA graphs is shown in Figure 10. The data provided by the tool is used for human review and reference, and is not used in prediction or any of the subsequent sections.

3.5 PLS Analysis

Partial Least Squares (PLS) Analysis also has an important role in multivariate analyses. PLS can be broadly described as an extension of the PCA method (Felipe-Sotelo et al., 2008). While PCA can be used to condense a large set of predictive variables, it does not take into account the impact of dependent variables. This makes PCA less suitable for studies involving a large number of predictive variables which involve different measures. PLS extracts values from both predictive (X) and dependent (Y) variables, and works to maximize the correlation between both (Hegazy et al., 2018). Hence, it ensures that components have greater correlation with dependent variables.

The current project involves over a hundred predictive variables. While most relate to raw material composition, they involve different measures (e.g. elements and vitamin composition). Thus, factoring dependent variables when creating components can enable a more accurate study.

As with PCA, Statistica provides an in-built tool to perform PLS analyses. Similarly, the analysis can be performed either through the interface, or using a script, of which the latter will be used, as with PCA. However, components calculated using the PLS analysis will be internally recorded, stored and used for further analysis by the Analysis Tool, including generating predictions. Component data is accessible in the spreadsheets.

```
newanalysisPLS[illegible].UseGlobalOutputSettings = False
With newanalysis[illegible].OutputOption
    .Placement = scAnalysisWorkbook
    .AutoPlaceResultInWorkbook = True
    .PlaceNewResultAtTop = True
    .ReportPlacement = scNoReport
    .ReportFont = "Courier New"
    .ReportFontSize = 9
    .SupplementaryInfoLevel = scInfoLevelNone
    .MSWordPlacement = scNoMSWord
End With

newanalysisPLS[illegible].Run


Dim oAD3PLS[illegible] As STAMSPC.PLSResults
Set oAD3PLS[illegible] = newanalysisPLS[illegible].Dialog
oAD3PLS[illegible].SortVariablesByImportance = True
oAD3PLS[illegible].TSquareControlLimit = 99
oAD3PLS[illegible].TSquareWarning = False
oAD3PLS[illegible].TSquareWarningLimit = 95
'oAD3PLS[illegible].SelectFirstListForX = True
'oAD3PLS[illegible].SelectSecondListForX = True
'oAD3PLS[illegible].SelectedFirstListOfComponents = "1"
'oAD3PLS[illegible].SelectedSecondListOfComponents = "2"
oAD3PLS[illegible].LabelWithNames = True
oAD3PLS[illegible].ComputeScoreLimitFromStdDev = True
oAD3PLS[illegible].ScoreControlLimit = 3
oAD3PLS[illegible].ScoreWarning = False
oAD3PLS[illegible].ScoreWarningLimit = 2
oAD3PLS[illegible].TypeOfControlChart = scMspcShewhartIndividual
oAD3PLS[illegible].UsingUserDefinedTarget = False
oAD3PLS[illegible].UsingUserDefinedStdDev = False
```

Figure 11: SVB Code to perform a PLS Analysis in Statistica.

The script used to generate a PLS analysis is shown in Figure 11. As with a PCA analysis, codes can be obtained using the "Record Macro" function, as described previously. Other settings, such as font and display settings, and also be adjusted in the script.

```
i = PLS[illegible]CheckSheet.NumberOfCases
' Loop term
j = 0
' Number of valid components
k = i
' Total Number of components
While i >0
    ii = PLS[illegible]CheckSheet.Cells(i,6).Value
    If ii > 0 Then
        j = i
        i = 0
    End If
    i = i - 1
Wend
m = k - j
' Number of Invalid Components
ComponentPLS[illegible] = j
If j = 0 Then
    ErrorActive = 100
    ErrorString = ErrorString + "Warning: No valid components detected for [illegible]"
    GoTo EscapePLS[illegible]
End If
If m > 0 Then
    While m > 0
        newanalysisPLS[illegible].Dialog.RemoveLastComponent
        k = k - 1
        oAD3PLS[illegible].NumberOfEigenvaluesToPlot = k
        m = m - 1
    Wend
End If
```

Figure 12: Elimination of non-significant PLS components.

Principal Components derived from the PLS analysis will be used in subsequent parts of the tool. Hence, it is imperative that only significant Principal Components are chosen.

Additional codes (shown in Figure 12) have been added to the script to screen all Principal Components generated by the analysis, and to remove those which are not significant. In the unlikely event that the set of variables are unable to generate even one significant Principal Component, a warning message is sent to the user, and the tool voids that set of data for subsequent analyses.

```
Set oStaDocsPLS[illegible] = oAD3PLS[illegible].LoadingSpreadsheet
Set AnalysisOutputPLS[illegible] = newanalysisPLS[illegible].RouteOutput(oStaDocsPLS[illegible])
If (AnalysisOutputPLS[illegible].HasWorkbook=True) Then
    Set w=AnalysisOutputPLS[illegible].Workbook
    Set wi=w.Root.Child
    While (wi.Type<>scWorkbookItemTypeSpreadsheet)
        Set wi=wi.Child
    Wend
    Set PLSXLoad[illegible] =wi.Extract(scWorkbookExtractCopy)
End If
Set oStaDocsPLS[illegible] = oAD3PLS[illegible].XWeightSpreadsheet
Set AnalysisOutputPLS[illegible] = newanalysisPLS[illegible].RouteOutput(oStaDocsPLS[illegible])
If (AnalysisOutputPLS[illegible].HasWorkbook=True) Then
    Set w=AnalysisOutputPLS[illegible].Workbook
    Set wi=w.Root.Child
    While (wi.Type<>scWorkbookItemTypeSpreadsheet)
        Set wi=wi.Child
    Wend
    Set PLSXWeight[illegible] =wi.Extract(scWorkbookExtractCopy)
End If

Set oStaDocsPLS[illegible] = oAD3PLS[illegible].XScoreSpreadsheet
Set AnalysisOutput[illegible] = newanalysis[illegible].RouteOutput(oStaDocsPLS[illegible])
If (AnalysisOutputPLS[illegible].HasWorkbook=True) Then
    Set w=AnalysisOutputPLS[illegible].Workbook
    Set wi=w.Root.Child
    While (wi.Type<>scWorkbookItemTypeSpreadsheet)
        Set wi=wi.Child
    Wend
    Set PLSXScore[illegible] =wi.Extract(scWorkbookExtractCopy)
End If
```

Figure 13: SVB Code to extract graphical information from a PLS Analysis.

Scripts used to call and generate different graphs are shown in Figure 13. These are meant for user reference. Only the extracted Principal Components will be used for the rest of the analysis.

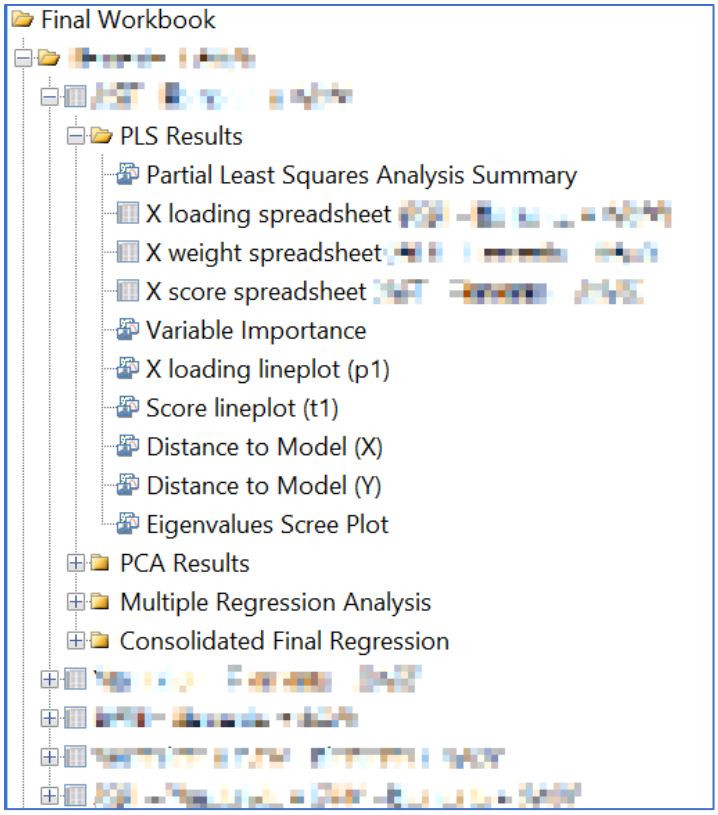


Figure 14: PLS Output data/graphs from Analysis Tool.

A subset of all available PCA graphs is shown in Figure 14. Data in the spreadsheets, as well as Principal Component values (stored separately) are stored in internal spreadsheets and arrays for subsequent parts of the analysis. Graphs provided by the tool are used for human review and reference, and are not carried forward to the subsequent sections.

An advantage of Statistica is that for each Principal Component, predictor variables which clearly have no variation with the dependent variable are automatically removed from the analysis. This is advantageous as some predictor variables do not vary at all, hence making them redundant to the overall analysis. While it is technically possible to individually apply every variable to the subsequent regression analysis, applying the PLS analysis streamlines the overall process significantly and eliminates redundant variables.

3.6 Multiple Regression Analysis

Statistica also provides tools for regression analyses. A Multiple Regression Analysis (Gordon et al., 2018) involves a single dependent variable, and multiple predictor variables. The analysis attempts to determine a correlation in the following form between the predictor and dependent variables.

$$Y = k_0 + k_1X_1 + k_2X_2 + \ldots$$

Where Y is the single dependent variable, X represents the range of predictor variables, the corresponding k values represent the coefficients of each X variable and k_0 represents a constant.

By working out the coefficients of the provided equation, it is possible to develop a mathematical relationship between the dependent and predictor variables. Statistica also provides the Coefficient of Determination (R^2), which measures the overall fit of the data to the statistical model. R^2 values of 0.6 or higher typically indicate a good fit between the model and the data, and the potential for the model to predict how process variables based on data provided.

```
If ComponentPLS[redacted] > 0 Then
    Dim newanalysisMRG[redacted] As Analysis
    Set newanalysisMRG[redacted] = Analysis(scMultipleRegression, [redacted]SortSheet)
    Dim oStaDocsMRG[redacted] As StaDocuments

    ' Multiple Linear Regression: [redacted]
    Dim oAD1MRG[redacted] As STARegression.RegStartup
    Set oAD1MRG[redacted] = newanalysisMRG[redacted].Dialog
    If Component[redacted] = 1 Then
        oAD1MRG[redacted].Variables = "[redacted] "
    End If
    If Component[redacted] = 2 Then
        oAD1MRG[redacted].Variables = "[redacted] "
    End If
    If Component[redacted] = 3 Then
        oAD1MRG[redacted].Variables = "[redacted] "
    End If
    If Component[redacted] = 4 Then
        oAD1MRG[redacted].Variables = "[redacted] "
    End If
    If Component[redacted] = 5 Then
        oAD1MRG[redacted].Variables = "[redacted] "
    End If
    oAD1MRG[redacted].InputFile = scRegRawData
    oAD1MRG[redacted].CasewiseDeletionOfMD = True
    oAD1MRG[redacted].PerformNonDefaultStepwiseAnalysis = False
    oAD1MRG[redacted].ReviewDescriptiveStatistics = False
    oAD1MRG[redacted].ExtendedPrecisionComputations = False
    oAD1MRG[redacted].BatchProcessingAndPrinting = False


    newanalysisMRG[redacted].UseGlobalOutputSettings = False
    With newanalysisMRG[redacted].OutputOption
```

Figure 15: SVB Code to generate a Multiple Regression Analysis by the number of Principal Components.

The script used to generate a multiple regression analysis is shown in Figure 15. The analysis adjusts itself based on the number of Principal Components generated in the previous PLS Analysis, and skips the data set if no Principal Components of sufficient significance were detected. The script is able to accommodate up to five Principal Components, and will ignore any additional Principal Components. However, this is more of a redundant measure as it is nearly impossible that a set of data will yield such a large number of significant Principal Components. No data sets provided from the previous project have yielded more than two significant Principal Components.

```
newanalysisMRGvcd.Dialog.ResultsVariables = "
Set oStaDocsMRGvcd = oAD3MRGvcd.ScatterplotOfObservedValues
Set AnalysisOutputMRGvcd = newanalysisMRGvcd.RouteOutput(oStaDocsMRGvcd)
If (AnalysisOutputMRGvcc.HasWorkbook=True) Then
    Set w=AnalysisOutputMRGvcd.Workbook
    Set wi=w.Root.Child.Child.Next
    While (wi.Type<>scWorkbookItemTypeGraph)
        Set wi=wi.Child
    Wend
    Set MRGvcdVersus =wi.Extract(scWorkbookExtractCopy)
End If

If ComponentPLS > 1 Then
    newanalysisMRGvcc.Dialog.ResultsVariables = "
    Set oStaDocsMRGvcd = oAD3MRGvcd.ScatterplotOfObservedValues
    Set AnalysisOutputMRGvcd = newanalysisMRGvcd.RouteOutput(oStaDocsMRGvcd)
    If (AnalysisOutputMRGvcd.HasWorkbook=True) Then
        Set w=AnalysisOutputMRGvcd.Workbook
        Set wi=w.Root.Child.Child.Next
        While (wi.Type<>scWorkbookItemTypeGraph)
            Set wi=wi.Child
        Wend
        Set MRGvcdVersus2 =wi.Extract(scWorkbookExtractCopy)
    End If
End If
```

Figure 16: SVB Code to generate Regression plots from a Multiple Regression Analysis

The script (shown in Figure 16) also generates graphs from the analysis, mainly to compare an individual predictor variable with the dependent variable on a single plot. This is for user review and is not used in the rest of the analysis.

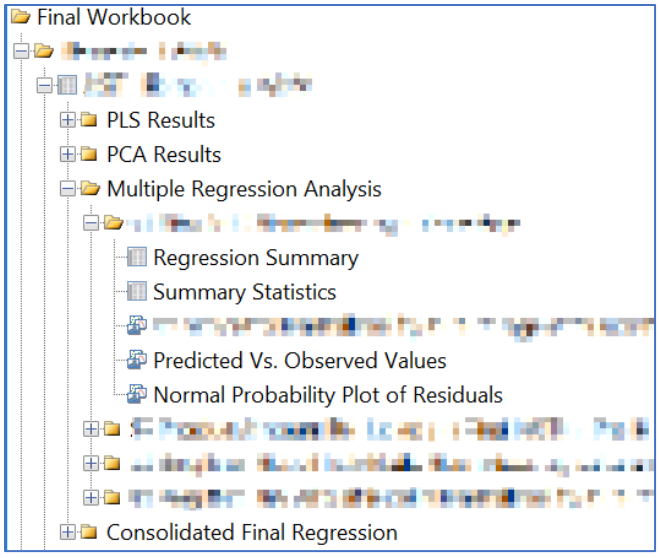


Figure 17: Multiple Regression Analysis Output graphs using Analysis Tool.

A subset of the data provided is shown in Figure 17. The regression summary and statistics provide key data to be used in the next step of the analysis.

	Regression Summary for Dependent Variable: [illegible] R= [illegible] R²= [illegible] Adjusted R²= [illegible] F([illegible] p<.[illegible] Std.Error of estimate: [illegible]					
N=[illegible]	b*	Std.Err. of b*	b	Std.Err. of b	t(22)	p-value
Intercept						
[illegible]						
[illegible]						
[illegible]						
[illegible]						

Figure 18: Multiple Regression Analysis – Predictor Variable Performance.

Data from the multiple regression is provided in the spreadsheet shown in Figure 18. This includes all Principal Components (significant) from the previous analysis and a few other process variables (such as pasteurization) which can be used as predictor variables. P-values (or probability values) provide an indication of the significance of the results. A p-value below 0.05 (Kennedy-Shaffer, 2019) indicates that the variable has managed to exceed a threshold in which it could be

considered significant, and will be used for subsequent analyses. Thus, all variables which fail to meet the threshold (p-value < 0.05) will be taken out, with the remainder carried forward to a second Multiple Regression analysis.

```
If ComponentPLS[redacted] = 1 Then
    If MRGvcdSummary[redacted].Cells(2,6).Value < 0.05 Then
        Entrypoint = Entrypoint + 1
        C1pvcd[redacted] = 1
    End If
    If MRGvcdSummary[redacted].Cells(3,6).Value < 0.05 Then
        Entrypoint = Entrypoint + 1
        PASpvcd[redacted] = 1
    End If
    If MRGvcdSummary[redacted].Cells(4,6).Value < 0.05 Then
        Entrypoint = Entrypoint + 1
        VIApvcd[redacted] = 1
    End If
    If MRGvcdSummary[redacted].Cells(5,6).Value < 0.05 Then
        Entrypoint = Entrypoint + 1
        VCDpvcd[redacted] = 1
    End If
End If
```

Figure 19: Consolidation of significant variables.

The b coefficients, the coefficients of the variables in the linear regression, will be taken from the second Multiple Regression Analysis, in which only significant variables will be considered. As shown in Figure 19, variables which do not meet the required p-value threshold are excluded. The remainder are put through the same analysis, and R^2 values as well as the b coefficients taken from the second multiple regression analysis are consolidated in the output workbook of the tool. This is applied to the next analysis.

3.7 Full Permutation Prediction

Various lots of feed and media are provided for the upcoming project, each of different compositions. By varying the selection and proportions of the provided lots, it is possible to vary the composition of elements, amino acids and vitamins in both process media and feed, albeit to an extent limited by the compositions of the base lots provided.

Optimizing the process variables essentially comes down to manipulating the selection and proportion of lots used to prepare the feed and media, such that the composition of each leads to an optimal result. Due to the large number of lots, and thus, the large number of variables involved, approaching the problem using a linear method is insufficient. Hence, a more comprehensive method is used, in which every possible permutation to combine lots is identified. The Analysis Tool performs a calculation for each, and returns the most ideal combinations.

```
i1 = 110
While i1 > 0
    i1 = i1 - 10
    i2 = 110
    While i2 > 0
        If CasNumElement[...] < 2 Then
            i2 = 10
        End If
        i2 = i2 - 10
        i3 = 110
        While i3 > 0
            If CasNumElement[...] < 3 Then
                i3 = 10
            End If
            i3 = i3 - 10
            i4 = 110
            While i4 > 0
                If CasNumElement[...] < 4 Then
                    i4 = 10
                End If
                i4 = i4 - 10
                iSum = i1 + i2 + i3 + i4
                If iSum = 100 Then
                    PermutationCount[...] = PermutationCount[...] + 1
                    Permutations[...](1,PermutationCount[...]) = i1
                    If CasNumElement[...] > 1 Then
                        Permutations[...](2,PermutationCount[...]) = i2
                    End If
                    If CasNumElement[...] > 2 Then
                        Permutations[...](3,PermutationCount[...]) = i3
                    End If
                    If CasNumElement[...] > 3 Then
                        Permutations[...](4,PermutationCount[...]) = i4
```

Figure 20: SVB Code to determine and compile all possible combinations of material lots.

The script shown in Figure 20 calculates all possible permutations (in increments of 10%) of a media or feed sample with up to four material lots available. Using a series of loops, it is possible to generate sets of data identifying every permutation.

	1	2	3
	Set [redacted] (%)	Set [redacted] (%)	Set [redacted] (%)
1	100	0	0
2	90	10	0
3	90	0	10
4	80	20	0
5	80	10	10
6	80	0	20
7	70	30	0
8	70	20	10
9	70	10	20
10	70	0	30
11	60	40	0
12	60	30	10
13	60	20	20
14	60	10	30
15	60	0	40
16	50	50	0
17	50	40	10
18	50	30	20
19	50	20	30
20	50	10	40
21	50	0	50
22	40	60	0
23	40	50	10
24	40	40	20
25	40	30	30

Figure 21: Spreadsheet of possible permutations generated by Analysis Tool.

An example of the permutation tool, based on three available lots, is shown in Figure 21. Every row indicates a possible permutation to organize the available lots of raw material to generate process feed or media.

Composition data is available for individual raw material lots, and the tool proportionally averages the data to determine the overall composition of the combined lots. These are subsequently used with the Principal Components in the PLS analysis to generate a component variable.

	R2 Values:			
	1 Set Number (Refer to Master Sheet)	2 VARIABLE 1	3 VARIABLE 2	4 VARIABLE 3
1	1			
2	2			
3	3			
4	4			
5	5			
6	6			
7	7			
8	8			
9	9			
10	10			
11	11			
12	12			
13	13			
14	14			
15	15			
16	16			
17	17			
18	18			
19	19			
20	20			

Figure 22: Output Spreadsheet of predicted performance of all combinations of raw material lots.

Using the component variable, any other relevant predictor variables, and the b coefficients obtained in the multiple regression analysis, the Analysis Tool can perform a full calculation to predict process variables for each permutation of raw material lot combinations. These variables can subsequently be sorted in descending order, allowing users to quickly identify lots or combinations with the most potential.

3.8 Limitations

The tool works under the assumption that feed and media compositions are the main contributor to process variables in the study. While data from the previous project suggests that this might be a reasonable conclusion, based on the strong correlations between feed and media compositions as well as process variables, more research needs to be done to determine if this can be reproduced.

Additionally, some sets of multiple regressions produced somewhat poor R^2 values, suggesting a tenuous correlation between some sets of process variables and predictor variables. Nevertheless, predicted process variables from these sets could simply be excluded from the study, and the tool provides users with R^2 values for each prediction up front, to determine the extent to which the data could be useful.

4. Conclusion

A full analytical tool was developed in SVB for the current project, capable of rapidly aggregating large quantities of data from previous project, and generating regression coefficients using Multiple Regression to be applied to future projects. Additionally, the Analysis Tool is able to aggregate raw material batch data from current projects and generate predictions for process variables under a wide range of possible permutations. From there, users can easily pick out ideal combinations of raw materials for feed and media to generate the ideal process results.

The Analysis Tool is designed to be as efficient and easy to use as possible, requiring a single push of the "Run" button and file selections to generate a wide range of result spreadsheets, where users can easily view data and determine ideal selections of raw materials.

Subsequent projects can also make use of the Analysis Tool, with slight modifications to be made in the event that there are changes to the required process variables, or any other possible changes. Due to the flexibility of the script, future projects can easily make use of a slightly modified tool if there are any changes to process variables or inputs. Most of the existing code structures can be retained while providing for new variables or methods of calculation. The script

is designed to search for errors and adjusts itself to datasets of various sizes. In its current state, it is an effective tool for the current project, and also has the potential to serve GlaxoSmithKline in subsequent studies.

5. References

Alin, A., Agostinelli, C., Gergov, G., Katsarov, P. and Al-Degs, Y. (2019). Robust multivariate diagnostics for PLSR and application on high dimensional spectrally overlapped drug systems. *Journal of Statistical Computation and Simulation 89(6)*, 966-984.

Amraei, A. and Niazi, A. (2018). Partial Least Square and Parallel Factor Analysis Methods Applied for Spectrophotometric Determination of Cefixime in Pharmaceutical Formulations and Biological Fluids. *Iranian Journal of Pharmaceutical Research 17(4)*, 1191–1200.

Felipe-Sotelo, M., Tauler, R., Vives, I. and Grimalt, J. (2008). Assessment of the environmental and physiological processes determining the accumulation of organochlorine compounds in European mountain lake fish through multivariate analysis (PCA and PLS). *Science of the Total Environment 404*, 148-161.

Gordon, D., Huang, W., Burns, D., French, R. and Bruckman, L. (2018). Multivariate multiple regression models of poly(ethylene-terephthalate) film degradation under outdoor and multi-stressor accelerated weathering exposures. *PLoS ONE 13(12),* 1-30.

Hegazy, M., Boltia, S., Fayed, A. and Musaed, A. (2018). Advanced chemometrics manipulation of UV-spectroscopic data for determination of three co-formulated drugs along with their impurities in different formulations using variable selection and regression model updating. *Molecular and Biomolecular Spectorscopy 202,* 359-367.

Kennedy-Shaffer, L. (2019). Before $p < 0.05$ to Beyond $p < 0.05$: Using History to Contextualize *p*-Values and Significance Testing. *The American Statistician 73*, 82-90.

Lewis, S., Holl, H., Long, M., Mallicote, M. and Brooks, M. (2018). Use of principle component analysis to quantitatively score the equine metabolic syndrome phenotype in an Arabian horse population. *PLoS ONE 13(7),* 1-14.

Rentala, S. and Anand, B. (2014). Exploring the Determinants of Export Performance of Indian Pharmaceutical Industry - A Quantile Regression Approach. *Journal of Contemporary Research in Management 9(4)*, 15-24.

Ruvalcaba-López, J., Córdova-Fraga, T., Rosa-Alvarez, G., Murillo-Ortiz, B., Martínez-Espinosa, J., Guzmán-Cabrera, R. and Bernal-Alvarado, J. (2019). Qualitative evaluation of ferritin in serum samples by Raman spectroscopy and principal component analysis. *Lasers in Medical Science 34*, 35-40.

Wang, Y., Si, Y., Huang, B. and Lou, Z. (2018). Survey on the Theoretical Research and Engineering Applications of Multivariate Statistics Process Monitoring Algorithms. *The Canadian Journal of Chemical Engineering 96*, 2073-2085.

Biography

Victor W H Chan was born in 1992 in Singapore.

In 2011, Victor enlisted in the Singapore Armed Forces. He was posted to the 5th Singapore Infantry Regiment as a Motorized Infantry Commander, and participated in a training mission in Brisbane, Australia. He passed out from the Armed Forces holding the rank of Sergeant.

In 2013, Victor began his undergraduate degree at the National University of Singapore, where he majored in Chemical Engineering. Victor was also offered admission to the University Scholars' Program, a multidisciplinary academic program which accepts less than 3% of the undergraduate cohort. During his undergraduate studies, he spent a summer in Ho Chi Minh City, Vietnam, performing research work in wastewater treatment. Additionally, he attended the University of Texas at Austin for a semester under a University-wide exchange program.

In 2017, Victor began his MSE at the Johns Hopkins University. He is currently a full-time Co-Op worker at GlaxoSmithKline, pursing a course in practical training under the sponsorship of the Institute of NanoBioTechnology (INBT) Co-Op Program.